11 La shell (1)

Si potrebbe dire, sbagliando, che la shell di Linux è un po' l'equivalente dell interprete di comandi di Windows (il vecchio prompt di MS Dos). A prima vista si assomigliano, e condividono alcuni concetti. Tuttavia, mentre nel Dos era presente una shell che offriva un insieme minimo di funzionalità presenti nella shell Unix (d'altra parte Dos offriva una frazione infinitesimale delle caratteristiche di Unix), in Windows l'interprete dei comandi è andato via via perdendo d'importanza, fino ad essere, nelle ultime versioni del sistema, uno strumento che serve solo a qualche nostalgico per eseguire vecchissimi programmi e a poco più (avete provato a muovervi tra le directory via Dos? Personalmente lo trovo assurdo!).

In Linux, al contrario, la shell è tuttora una componente centrale e di vitale importanza, che difficilmente potrà essere completamente sostituita da un'interfaccia grafica, data la sua flessibilità. Purtroppo è anche un programma non facile da usare in tutte le sue potenzialità, e per questo dedicheremo due lezioni ad apprenderne i concetti base (abbastanza comunque da apprezzarne l'utilità).

Una shell è un programma intimamente legato ad un **terminale**, vale a dire una componente che permette input (da tastiera) ed output (per esempio a video). Su un unico computer possono coesistere più terminali (un po' come abbiamo visto che possono esistere più Desktop virtuali), e ci sono essenzialmente due modi di aprirne uno nuovo.
Avete a disposizione sei terminali testuali premendo (contemporaneamente) i tasti [CTRL][ALT] ed uno tra [F1] ed [F6]. [CTRL][ALT][F7] invece vi permetterà di tornare in ambiente grafico da un terminale testuale. Vi dovrebbe apparire una schermata simile a quella di Figura 1.

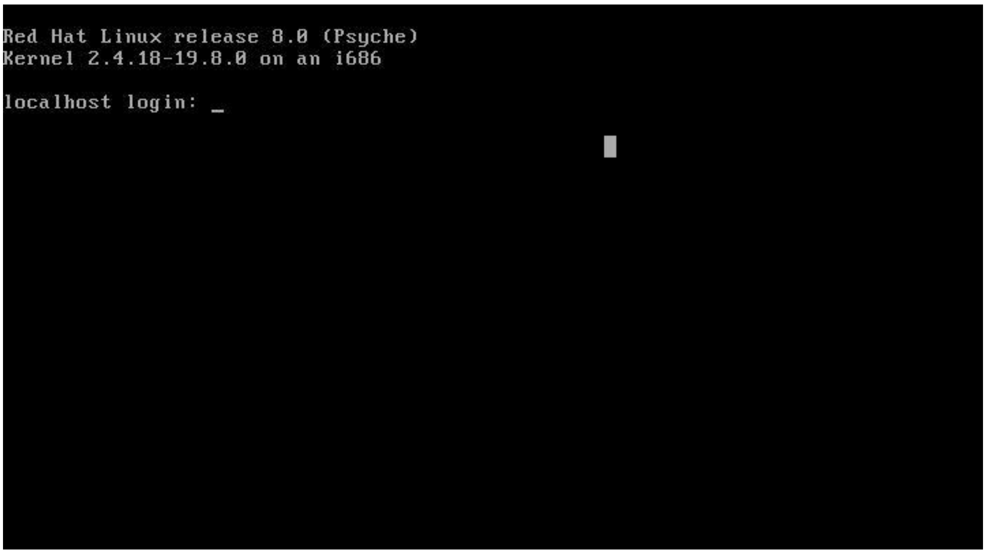


Figure 1: Terminale testuale

A questo punto basta dare login e password, e sarete al cospetto di una shell! Tuttavia, per motivi didattici, sarà meglio tornare in ambiente grafico ([CTRL][ALT][F7], ricordo) ed aprire un emulatore di terminale grafico. Potete fare click con il tasto destro del mouse sullo sfondo, e selezionare la voce *"Nuovo terminale"* (GNOME), oppure cercare l'applicazione *"Terminal"* nel menù dei programmi (tasto in basso a sinistra, con il cappello rosso), nel menù *"Strumenti di sistema"* (GNOME e KDE). Si dovrebbe aprire una finestra come quella mostrata in Figura 2.

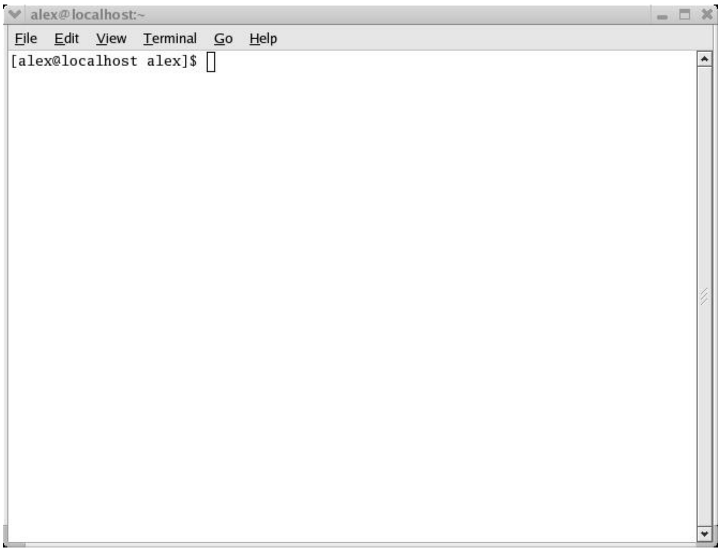


Figura 2: Terminale grafico

A questo punto è possibile inserire dei comandi, che saranno interpretati ed eseguiti.
Quali comandi abbiamo a disposizione? Sono veramente tanti, e per comprendere tutto, abbiamo bisogno di alcune nozioni di base.

In effetti, la shell è la versione a basso livello della finestra che possiamo aprire via interfaccia grafica, facendo click sulla nostra home directory (l'icona con la casetta sullo sfondo del desktop), come si vede in Figura 3.
Quando apriamo tale finestra, ci troviamo all'interno della cartella /home/nomeUtente, come si può vedere dal campo 'posizione', e vediamo una lista di tutti i files e directory (o cartelle, equivalente

mente) che vi sono contenute.

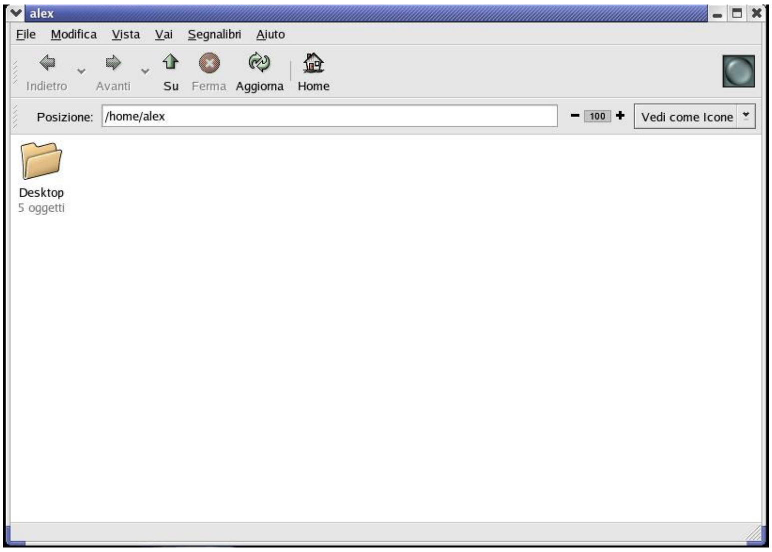


Figura 3: La nostra home directory

Allo stesso modo, quando usiamo la shell ci troviamo in ogni momento all'interno di una **directory**, cioè una cartella. Possiamo vederne il contenuto con il comando "ls" (esiste anche l'equivalente "dir", di sapore vagamente Microsoft), e possiamo sapere il nome della directory in cui ci troviamo con il comando "pwd" (che sta per Print Work Dir).
All'inizio ci troviamo nella nostra home directory, che ha nome /home/nomeUtente (nel mio caso è /home/alex), la stessa che abbiamo visualizzato graficamente, ed il listato che riceviamo dovrebbe essere l'equivalente testuale di tutte le icone.
I colori ci comunicano qualche informazione sul tipo di files, anche se non immediate come delle icone (se il vostro listato apparisse privo di colori, provate a digitare 'ls --color con due – prima di color): le directory (o cartelle) dovrebbero essere visualizzate in colore blu.

Con l'interfaccia grafica, ci possiamo spostare in altre cartelle facendoci click due volte sopra, e possiamo entrare nella cartella che contiene quella attuale premendo sul bottoncino della "Freccia verso l'alto" (le directory hanno una struttura gerarchica e, tranne la radice, sono contenute in qualche altra directory, e possono contenere altre directory), oppure mettendo nel campo posizione il nome per esteso della directory, se lo conosciamo (provate a sostituire la directory attuale con / e a premere [Invio]).
Nella shell, ci possiamo spostare con il comando "cd nome_directory", dove nome_directory va sostituito con la cartella in cui vogliamo entrare. Ad esempio, scrivendo "cd /", ci sposteremo nella directory radice, che contiene tutte le altre. Con un "ls" a questo punto, possiamo osservare la struttura standard delle directory Linux: abbiamo le già citate **etc**, **usr** e **home**, più numerose altre quali:

- bin, che contiene programmi di sistema,
- dev, che contiene dei puntatori a tutte le periferiche ed i dischi del vostro computer,
- var, che contiene informazioni che (di solito) vengono generate automaticamente da Linux, come file di log e cache.

Possiamo cambiare ancora una volta directory per andare, per esempio, all'interno di usr, con il comando "cd usr" seguito, come al solito, da [Invio].
A questo punto, per tornare nella nostra home directory, possiamo digitare "cd /home/nomeUtente".

Notate che il nome della vostra directory è la concatenazione del **percorso** che bisogna fare per raggiungerla, quasi come se fosse un indirizzo internet: si parte da /, si va in home, e quindi in nomeUtente. Il secondo slash (/) è un separatore di nome directory, e va messo tra i nomi delle cartelle che compongono un percorso (ad esempio /pippo/pluto/paperino/qui/quo/qua è il percorso di una directory o programma di nome qua, contenuto in una directory di nome quo, a sua volta dentro qui, e così via fino alla radice).
Tornando in usr (che dalla vostra home directory può quindi essere raggiunta con "cd /usr", potevamo anche seguire un altro percorso. In ogni directory sono contenute sempre due directory speciali:

- . (proprio un punto), che è la directory stessa
- .. (doppio punto) che è la directory che contiene la directory in cui ci troviamo

Ossia, se scriviamo "cd ." e diamo [Invio], ci troveremo nella stessa directory senza muoverci (provare per credere!), mentre se digitiamo "cd .." seguito dal solito [Invio], andremo a finire nella directory che contiene quella in cui stiamo (l'unica eccezione è la directory /, che non essendo contenuta in nessuna directory torna sempre in se stessa). Dunque, da usr, possiamo scrivere "cd .." (tralascerò il tasto [Invio] d'ora in poi, che andrà premuto dopo ogni comando), andando a finire nella directory / (che infatti la conteneva). A questo punto, con "cd home" e "cd nomeUtente", ci ritroviamo nella home. Ma potevamo anche fare tutto in un passo unico con "cd ../home/nomeutente"!
So che ciò può risultare confuso per un principiante, ma è solo questione di pratica (sono inoltre quasi certo che avete già usato qualche volta l'interprete dei comandi di Windows, e siete pratici).

Nell'interfaccia grafica si può creare una nuova cartella con un semplice click del tasto destro, scegliendo "Nuova cartella", e decidendo un nome per questa, e la si può cancellare trascinandola sul cestino.
Nella shell, per creare una nuova directory dovete usare il comando "mkdir nome_directory" e per cancellare una directory vuota con il comando "rmdir nome_directory" (per cancellare un file invece il comando da digitare è "rm nome_del_file"). Per cancellare una directory con tutto il suo contenuto (files e sottodirectory), si deve digitare il comando "rm –r nome_directory", ma dovete essere assolutamente sicuri di quello che state facendo, perché questa è un'operazione senza ritorno (non esiste il cestino dei rifiuti nel duro mondo della shell!).
Attenzione: se non siete entrati come utente root (e non dovreste essere entrati così, ma come il semplice utente che DOVRESTE aver creato durante l'installazione), potrete creare directory solo nella vostra home directory o in /tmp (che è una directory in cui tutti possono scrivere e leggere, ma è svuotata ad ogni avvio di Linux, ed è quindi adatta alla creazione di soli file e directory temporanei, come dice il nome).
Provate a tornare nella vostra home (una curiosità: esistono altri due modi per tornarvi, vista la sua grande importanza per ogni utente, e sono "cd" seguito da [Invio] e "cd ~nomeUtente" anch'esso seguito da [Invio], che vi permette di entrare nella home di qualsiasi utente tranne root - se ce ne fossero altri - se ne avete il permesso), e a vedere cosa c'è all'interno con "ls". Ci dovrebbe essere solo la directory Desktop. Ne potete creare una nuova con "mkdir pippo", e la potete vedere con un nuovo "ls". Potete entrarci con "cd pippo", e da li potete tornare nella vostra home con "cd ..", o "cd /home/nomUtente". Cancellatela pure con "rmdir pippo". Provate ad andare nella radice, con "cd /", e a scrivere "mkdir pippo". Se non siete root, dovrebbe apparire un errore, perché non ne avete il diritto.

Ogni file ed ogni directory, infatti, appartengono a qualche utente, e hanno diritti di lettura, scrittura o esecuzione (accesso per le directory) che possono essere solo per l'utente cui appartengono, per tutto il suo gruppo (potete ignorare questo concetto), o per tutti. Si possono vedere i diritti di un file con un'opzione del comando ls. Dalla directory radice, provate a scrivere "ls -l", e vi verrà fuori un

listato di directory molto dettagliato (Figura 4).
La prima lettera di ogni riga ci dice che cosa è quell'entrata: 'd' sta per directory, '-' per file (nel nostro caso sono tutte directory).
Segue poi la lista dei permessi del file, che è una sequenza di 9 caratteri. Si leggono a gruppi di tre, ed hanno il seguente significato:

- i primi tre sono i permessi per il proprietario del file,
- i secondi tre sono i permessi per il gruppo a cui lui appartiene,
- gli ultimi tre sono i permessi per il resto degli utenti.

I tre caratteri, in ordine, sono di lettura, di scrittura, e di esecuzione, e sono abilitati, rispettivamente, se è presente una 'r' (read), una 'w' (write), ed una 'x' (o talvolta una 't') (eXecute). Un segno '-' vuol dire che non è abilitato que permesso. Il nome del proprietario del file compare nella terza riga, quello del gruppo nella quarta.

```
alex@localhost:/
File  Edit  View  Terminal  Go  Help
CD%2fDVD-ROM  Cestino  Floppy  Home  Start Here
[alex@localhost Desktop]$ cd /
[alex@localhost /]$ mkdir pippo
mkdir: impossibile creare la directory `pippo': Permission denied
[alex@localhost /]$ ls -l
totale 196
drwxr-xr-x    2 root     root         4096 gen 12 11:47 bin
drwxr-xr-x    3 root     root         4096 gen 12 21:32 boot
drwxr-xr-x   20 root     root       118784 gen 13 20:12 dev
drwxr-xr-x   56 root     root         4096 gen 13 20:12 etc
drwxr-xr-x    4 root     root         4096 gen 12 12:13 home
drwxr-xr-x    2 root     root         4096 giu 21  2001 initrd
drwxr-xr-x    7 root     root         4096 gen 12 11:38 lib
drwx------    2 root     root        16384 gen 12 11:34 lost+found
drwxr-xr-x    2 root     root         4096 ago 27 06:49 misc
drwxr-xr-x    4 root     root         4096 gen 12 12:02 mnt
drwxr-xr-x    2 root     root         4096 ago 23  1999 opt
dr-xr-xr-x   68 root     root            0 gen 13 21:12 proc
drwxr-x---    6 root     root         4096 gen 13 20:22 root
drwxr-xr-x    2 root     root         8192 gen 12 11:38 sbin
drwxrwxrwt   13 root     root         4096 gen 13 21:27 tmp
drwxr-xr-x   15 root     root         4096 gen 12 11:35 usr
drwxr-xr-x   17 root     root         4096 gen 12 11:42 var
[alex@localhost /]$
```

Figura 4: Listato esteso

Facciamo un paio di esempi per capire meglio.
L'entrata tmp è una directory, perché il primo carattere è una lettera 'd', e ha i seguenti diritti: rwx rwx rwt. Il proprietario è l'utente root.
I primi tre caratteri vogliono dire che root stesso può leggere la directory, ci può scrivere e ci può accedere. I secondi tre caratteri vogliono dire che gli utenti nel gruppo di root (chiamato a sua volta root), possono fare le stesse cose. La terza tripletta significa che anche il resto degli utenti ha gli stessi diritti.
Ed infatti, come già detto, tmp è una directory in cui tutti possono scrivere, leggere ed accedere liberamente.
La directory lost+found invece, è sempre di root, ma ha i seguenti permessi: rwx --- ---. Cioé: solo root può farci qualcosa. Infatti, scrivemdo "cd lost+found" ci vedremo restituire un errore, non essendo root.

Ci sono dei modi per cambiare i permessi di un file che si possiede, ma questo fa parte di un corso

un pochino più avanzato!

Dall'interfaccia grafica, possiamo **eseguire** dei programmi situati nella cartella che stiamo visualizzando, che non sono altro che dei files con i diritti di esecuzione abilitati.
Mettiamo per esempio nel campo 'Posizione' la directory '/usr/X11R6/bin' (o portiamoci in tale directory con una serie di passi). Cerchiamo tra la lista di files quello chiamato 'xeyes', e facciamoci un doppio click sopra. Questo è un semplicissimo programmino che fa apparire due occhietti che seguono il vostro mouse sullo schermo (Figura 5). Per chiuderlo, potete fare click con il mouse sull'icona di chiusura posta in alto a destra della finestra di xeyes.

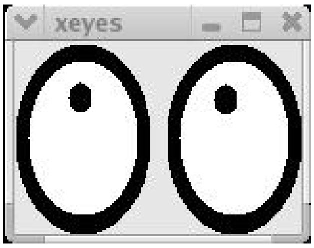


Figura 5: xeyes

Per eseguire un comando da shell, basta scriverne il nome seguito da [Invio]. Se Linux è ben configurato, dovreste già avere nel PATH tutte le directory necessarie. Proviamo ad esempio, a scrivere "xeyes" (tutti i comandi verranno da me scritti tra virgolette, che voi non dovete scrivere), e dare invio. Lo stesso programma di prima dovrebbe apparire.
Per chiuderlo, oltre allo stesso metodo descritto in precedenza, potete premere [CTRL][c] sulla shell, che come avrete notato si era fermata in attesa che il programma terminasse.

Invece che xeyes avreste potuto scrivere un qualsiasi comando quale "mozilla" (un web browser), "evolution" (un cliente di posta elettronica), o addirittura "xterm" (un altro terminale grafico, che si aggiungerebbe al primo!). E' tuttavia spesso molto più comodo cercare un programma tra la lista di quelli nel menù generale (anche se per esempio xeyes non è presente, pur essendo installato).
Uno strumento estremamente utile della shell (che personalmente non ho mai capito perché non è stato copiato da MS Dos ai suoi tempi), è il tasto **[TAB]** (quello accanto alla Q a sinistra, se non lo conosceste con questo nome), che avvia il completamento dei nomi dei programmi e delle directory.
Provate ad esempio a scrivere nella shell "mozi", e a premere [TAB] (ovviamente prima di [Invio]). Come per magia, il nome sarà completato in "mozilla", e voi potrete dare [Invio] ed avviare il browser.
Provate invece a scrivere "psm", e a premere [TAB]. Un beep vi avviserà che non è possibile completare il comando. Ciò può avvenire per due motivi:

- non c'è nessun comando il cui nome inizi per psm
- ce n'è più di uno, e il sistema non sa quale scegliere

Premendo una seconda volta [TAB], se siamo nel primo caso, un ulteriore beep ci avvertirà che non esiste nessun comando. Se siamo invece nel secondo caso, verranno visualizzate le possibili opzioni (se sono veramente tante ci verrà chiesto se le vogliamo davvero vedere). Nel nostro caso specifico ci vengono presentati due nomi, tra cui possiamo scegliere quello che volevamo.
Cancellando la linea che avevamo scritto, possiamo premere [TAB] senza che niente sia stato scritto. Ciò equivale a chiedere al sistema i nomi di TUTTI i programmi che possiamo eseguire. Ci verrà chiesto (in inglese, essendo la traduzione del sistema non ancora perfetta) se vogliamo davvero visualizza i numerosi file (nel mio computer sono 1866, nel vostro potrebbero essere un numero differente). Premendo 'y', vi sarà presentata una lista di programmi ordinati alfabeticamente, dai segni di interpunzione alla z: potete scorrerla premendo [Invio] per andare avanti riga per riga, o [Spazio] per cambiare pagina, o potete uscire in qualsiasi momento alla shell premendo [q]).

Anche nel caso delle directory esiste il completamento dei nomi: provate dalla vostra home directory a scrivere "cd /" e a premere [TAB] (due volte, perché esiste più di una directory in /): vi apparirà una lista di tutte le directory contenute in /. Provate ora ad aggiungere una "u", e a premere [TAB] una sola volta: ecco riapparire il nome usr, che è l'unica directory il cui nome inizia per u presente in /.

Un ultimo concetto che vorrei esporre è quello dell'accesso ai cd ed ai dischetti. Mentre in Windows basta provare a premere sull'icona del lettore cd o floppy per dire al sistema di provare a leggerli (dando errore qualora non avessimo inserito il disco), in Linux le cose potrebbero essere diverse.
Di norma quando si usa Gnome o KDE, inserendo un cd si dovrebbe aprire immediatamente una finestra con il contenuto del disco.
KDE inoltre ha sullo sfondo le icone del cd e del dischetto, che si comportano esattamente come in Windows.
Nello Gnome, una volta inserito un cd (o un dischetto), si può premere con il tasto destro del mouse sullo sfondo del desktop, selezionare la voce "Dischi"->"CD-ROM" (o "Dischetto") per far apparire l'icona corrispondente, su cui potremo fare click per esaminarne il contenuto.
Quando non abbiamo più bisogno del cd o del dischetto, possiamo premere con il tasto destro sopra l'icona, e selezionare "Espelli" per il cd, o "Smonta" per il floppy.
Può tuttavia accadere (raramente) che ciò non funzioni: è necessario agire da shell.
Il lettore cd ed il lettore dischetti sono dei "device", cioè dei files speciali contenuti nella directory /dev. Tali files sono chiamati cdrom e fd0 (se avete più cd, saranno cdrom, cdrom1, cdrom2, etc).
Per poterci accedere, bisogna montare il device, cioè fare in modo che una directory "regolare" punti al contenuto del device: questo si ottiene con il comando

mount /dev/cdrom /mnt/cdrom

nel caso del lettore cd, e

mount /dev/fd0 /mnt/floppy

nel caso di lettore dischetto.
A questo punto, potete usare la directory /mnt/cdrom (o /mnt/floppy) come una normale directory, copiando dei files, leggendoli o eseguendoli (in un esercizio vedremo come fare).
Per smontarli, bisogna digitare il comando
umount /mnt/cdrom (o umount /mnt floppy).

Abbiamo dunque imparato ad eseguire comandi e a spostarci tra le directory da shell (oltre che parecchie nozioni di base).
Nella prossima lezione impareremo cosa sono i processi e come si gestiscono, e qualche trucchetto in più per sfruttare la shell al meglio.